

This Page Is Inserted by IFW Operations
and is not a part of the Official Record

BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images may include (but are not limited to):

- BLACK BORDERS
- TEXT CUT OFF AT TOP, BOTTOM OR SIDES
- FADED TEXT
- ILLEGIBLE TEXT
- SKEWED/SLANTED IMAGES
- COLORED PHOTOS
- BLACK OR VERY BLACK AND WHITE DARK PHOTOS
- GRAY SCALE DOCUMENTS

IMAGES ARE BEST AVAILABLE COPY.

**As rescanning documents *will not* correct images,
please do not report the images to the
Image Problems Mailbox.**

PATENT ABSTRACTS OF JAPAN

(11)Publication number : 2000-196468

(43)Date of publication of application : 14.07.2000

(51)Int.Cl.

H03M 13/23

(21)Application number : 10-366630

(71)Applicant : NEC IC MICROCOMPUT SYST LTD

(22)Date of filing : 24.12.1998

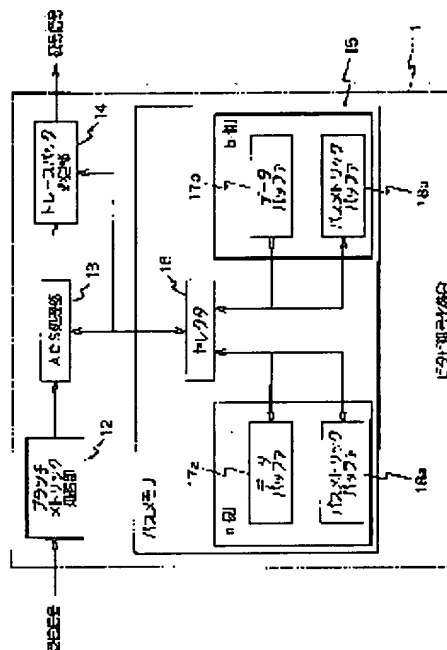
(72)Inventor : MATSUBARA TAKETSUGU

(54) VITERBI DECODER AND BITERBI DECODING METHOD

(57)Abstract:

PROBLEM TO BE SOLVED: To prevent deterioration in processing performance due to the number of times of memory accesses to a path memory in the case of trace-back processing and to reduce a capacity of the path memory.

SOLUTION: An ACS processing section 13 estimates a most probable path in each state at a present time on the basis of a path metric calculated by branch metric processing for each unit time and a path metric up to a preceding time stored in either of a couple of path metric buffers 18a, 18b, rearranges maximum likelihood decoding information up to the preceding time stored either of a couple of data buffers 17a, 17b on the basis of the estimated path and writes the rearranged information to the other, updates the path metric stored in either of a couple of the path metric buffers 18a, 18b and writes the updated path metric to the other. A trace back processing section 14 retrieves decoded data on the basis of the path metric stored up to a present time stored in the path metric buffers 18a, 18b and the maximum likelihood decoding information up to the present time stored in the data buffers 17a, 17b.



LEGAL STATUS

[Date of request for examination] 24.12.1998

[Date of sending the examiner's decision of rejection]

[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]

[Date of final disposal for application]

[Patent number] 3260714

[Date of registration] 14.12.2001

[Number of appeal against examiner's decision of rejection]

[Date of requesting appeal against examiner's]

decision of rejection]

[Date of extinction of right]

Copyright (C); 1998,2003 Japan Patent Office

(19) 日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11) 特許出願公開番号

特開2000-196468

(P2000-196468A)

(43) 公開日 平成12年7月14日 (2000.7.14)

(51) Int.Cl.⁷

H 0 3 M 13/23

識別記号

F I

H 0 3 M 13/12

テーマコード(参考)

5 J 0 6 5

審査請求 有 請求項の数 6 O L (全 12 頁)

(21) 出願番号 特願平10-366630

(22) 出願日 平成10年12月24日 (1998.12.24)

(71) 出願人 000232036

日本電気アイシーマイコンシステム株式会
社

神奈川県川崎市中原区小杉町1丁目403番
53

(72) 発明者 松原 岳次

神奈川県川崎市中原区小杉町一丁目403番
53 日本電気アイシーマイコンシステム株
式会社内

(74) 代理人 100079164

弁理士 高橋 勇

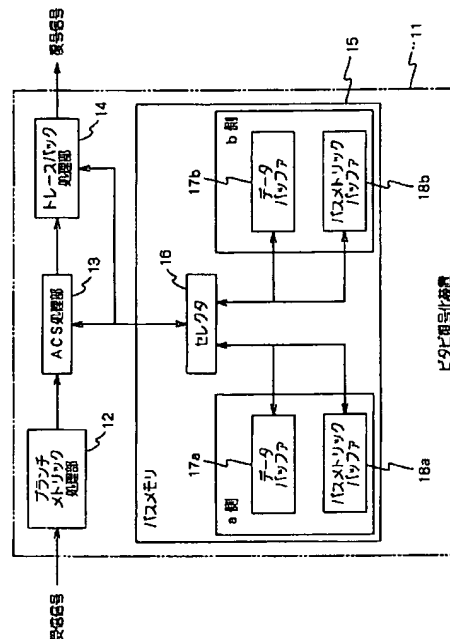
Fターム(参考) 5J065 AA03 AB01 AC02 AD10 AE02
AF01 AF03 AG05 AH06 AH23

(54) 【発明の名称】 ビタビ復号化装置およびビタビ復号化方法

(57) 【要約】

【課題】 トレースバック処理時におけるバスメモリへのメモリアクセス回数により処理性能が低下することを防止し、また、バスメモリの容量を低減する。

【解決手段】 ACS処理部13は1単位時間毎に、ブランチメトリック処理により計算されたブランチメトリックと1対のバスメトリックバッファ18a、18bの一方に記憶された前時刻までのバスメトリックに基づいて、現時刻の各状態における最も確からしいバスを推定して、このバスに基づいて1対のデータバッファ17a、17bの一方に記憶された前時刻までの最尤復号情報を並び替えて他方に書き込むとともに、1対のバスメトリックバッファ18a、18bの一方に記憶されたバスメトリックを更新して他方に書き込む。トレースバック処理部14はバスメトリックバッファ18a、18bに記憶された現時刻までのバスメトリックとデータバッファ17a、17bに記憶された現時刻までの最尤復号情報に基づいて復号データを検索する。



【特許請求の範囲】

【請求項 1】 前時刻から現時刻までの 1 単位時間に遷移可能な全てのバスについてブランチメトリックを計算するブランチメトリック処理手段と、

前時刻までの最尤復号情報と現時刻までの最尤復号情報をそれぞれ記憶するための 1 対のデータバッファと、前時刻までのバスメトリックと現時刻までのバスメトリックをそれぞれ記憶するための 1 対のバスメトリックバッファと、

1 単位時間毎に、前記ブランチメトリック処理手段により計算されたブランチメトリックと前記 1 対のバスメトリックバッファの一方に記憶された前時刻までのバスメトリックに基づいて、現時刻の各状態における最も確からしいバスを推定して、このバスに基づいて前記 1 対のデータバッファの一方に記憶された前時刻までの最尤復号情報を並び替えて他方に書き込むとともに、前記 1 対のバスメトリックバッファの一方に記憶されたバスメトリックを更新して他方に書き込む ACS 処理手段と、前記バスメトリックバッファに記憶された現時刻までのバスメトリックと前記データバッファに記憶された現時刻までの最尤復号情報に基づいて復号データを検索するトレースバック処理手段と、

を有するビタビ復号化装置。

【請求項 2】 前記データバッファには、最尤復号情報として各バスの最尤復号データ候補が記憶されることを特徴とする請求項 1 記載のビタビ復号化装置。

【請求項 3】 前記データバッファには、最尤復号情報として各バスの最尤状態遷移履歴が記憶されることを特徴とする請求項 1 記載のビタビ復号化装置。

【請求項 4】 前時刻から現時刻までの 1 単位時間に遷移可能な全てのバスについてブランチメトリックを計算するブランチメトリック処理ステップと、

1 単位時間毎に、前記ブランチメトリック処理ステップにより計算されたブランチメトリックと 1 対のバスメトリックバッファの一方に記憶された前時刻までのバスメトリックに基づいて、現時刻の各状態における最も確からしいバスを推定して、このバスに基づいて 1 対のデータバッファの一方に記憶された前時刻までの最尤復号情報を並び替えて他方に書き込むとともに、1 対のバスメトリックバッファの一方に記憶されたバスメトリックを更新して他方に書き込む ACS 処理ステップと、前記バスメトリックバッファに記憶された現時刻までのバスメトリックと前記データバッファに記憶された現時刻までの最尤復号情報に基づいて復号データを検索するトレースバック処理ステップと、

を有するビタビ復号化方法。

【請求項 5】 前記データバッファには、最尤復号情報として各バスの最尤復号データ候補が記憶されることを特徴とする請求項 4 記載のビタビ復号化方法。

【請求項 6】 前記データバッファには、最尤復号情報

として各バスの最尤状態遷移履歴が記憶されることを特徴とする請求項 4 記載のビタビ復号化方法。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】本発明は、トレースバック法を用いたビタビ復号化装置およびビタビ復号化方法に関し、特にトレースバック時間を短縮してビタビ復号化を高速化することができるビタビ復号化装置およびビタビ復号化方法に関する。

【0002】

【従来の技術】通信分野において、誤り訂正技術は非常に重要な技術の一つである。誤り訂正技術の一例として、ビタビ復号は畳み込み符号に対する最尤復号法の一般的なアルゴリズムとして知られており、誤り訂正能力が高いことから伝送誤りが生じ易い通信経路をもつ伝送方式における復号方式として、専用 LSI やデジタルシグナルプロセッサ (DSP) で構成されたビタビ復号装置が多く使用されている。また、近年では、マイクロプロセッサの高性能化、高機能化はめざましく、従来、専用 LSI などのハードウェアで行っていたビタビ復号化処理をソフトウェア (高性能マイクロプロセッサ+ソフトウェア) で実現することが可能になってきている。

【0003】ここで、問題になるのが処理性能である。従来の専用 LSI のように、CPU からのコマンドにより専用 LSI 単体 (シングルタスク) で処理していたのに比べ、マイクロプロセッサによるソフトウェア処理では、多くの場合、OS 実行などによるマルチタスク環境で使用されるため、マイクロプロセッサのリソースを 100% 占有することは難しく、従来の処理性能以上の高速性が求められている。

【0004】図面を参照しながら、従来のビタビ復号化装置およびビタビ復号化方法を説明する。図 9 は畳み込み符号化装置を示すブロック図、図 10 は図 9 の装置の状態を示す説明図、図 11 は図 9 の装置の状態遷移を示す説明図、図 12 は図 9 の装置のトレリス線図を示す説明図、図 13 は図 9 の装置の状態遷移の一例を示す説明図、図 14 は従来のビタビ復号化装置を示すブロック図、図 15 は図 14 のバスメモリを詳細に示すブロック図、図 16 はトレースバック法による復号動作を示す状態遷移を示す説明図、図 3 はブランチメトリック処理を示すフローチャート、図 17 は ACS (Add Compare Selection) 処理を示すフローチャート、図 18 はトレースバック処理を示すフローチャートである。

【0005】最初に、ビタビ復号化と対をなす畳み込み (トレリス) 符号化のアルゴリズムについて説明する。図 9 は説明を簡単にするために、拘束長 3、符号化率 1/2 の畳み込み符号化を行う符号化装置を示している。畳み込み符号化装置 81 は、3 ビットシフトレジスタ R1 (入力 in から順にビット「1」、「2」、「3」とする) と 2 つの排他的論理和回路 C1、C2 とで構成さ

れている。排他的論理和回路C1は、3ビットシフトレジスタR1の全ビット「1」、「2」、「3」についての排他的論理和を計算して出力信号out1として出力し、排他的論理和回路C2は、3ビットシフトレジスタR1のビット「1」とビット「3」についての排他的論理和を計算して出力信号out2として出力する。

【0006】ここで、3ビットシフトレジスタR1のビット「1」に蓄積されている情報ビットの値をa、ビット「2」に蓄積されている情報ビットの値をbとし、3ビットシフトレジスタR1に新たに1ビットの情報が入力されたとなると、3ビットシフトレジスタR1に蓄積されるビットの値は、ビット「1」から順に、入力された1ビットの情報、情報ビットa、bとなり、1ビットの情報ビットが入力されたときの畳み込み符号化装置81の状態数Xは、(a, b)の組み合わせ(X=0, 1, 2, 3)となる。この場合、図10に示すように

s0 = (0, 0)

s1 = (0, 1)

s2 = (1, 0)

s3 = (1, 1)

の4種類の状態が存在することになる。

【0007】一般に、拘束長がnであるときの状態数Xは、 $2^{(n-1)}$ (nはべき乗を示す)で表すことができる。図11は、現在の状態と、入力情報ビット(in)と、出力(out2, out1)と、遷移する状態の関係を示したものである。例えば、状態s0(a=b=0)のとき、入力情報ビットとして「0」が入力すると

(out2, out1) = (0, 0)

が出力され、状態s0に遷移する(状態は変わらない)。

【0008】同様に、状態s0(a=b=0)のとき、入力情報ビットとして「1」が入力すると(out2, out1) = (1, 1)が出力され、状態s1に遷移することがわかる。すなわち、畳み込み符号化装置81の畳み込み符号化では1ビットの入力情報ビットに対し、2ビットの符号ビット(符号化率1/2)を出力することになる。時刻tn(n=1, 2, 3...)における入力情報ビットに対し、以上の動作を繰り返すことにより、符号化を行うのが畳み込み符号化のアルゴリズムである。

【0009】図12は畳み込み符号化装置81で畳み込み符号化を行ったときの状態遷移を示した図(トレリス線図と呼ばれる)である。入力情報ビットとして「0」が入力された場合を実線で、入力情報ビットとして「1」が入力された場合を点線で、それぞれ示してある。このようにして符号化されたデータは、ランダムエラーの訂正能力が優れており、例えばビタビ復号化により最も確からしいパス(最尤パス)を決定して復号データに変換することで信頼性の高いデータを取得することができる。

【0010】次に、畳み込み符号化装置81により符号化されたデータをビタビ復号化装置131でビタビ復号化(トレースバック法)を行う場合のアルゴリズムについて説明する。ここで、畳み込み符号化装置81の初期状態を状態s0(a=b=0)とし、(1, 1, 0, 0, 0, 1, 1)の順で情報ビットが入力されたとなると、畳み込み符号化装置81は図13の実線で示される状態で遷移して符号化を行う。具体的には、畳み込み符号化装置81は最初の情報ビット=1が入力されると、符号化を行って2ビットの符号化データ(1, 1)を出力する。各入力情報ビットに対し、以上の処理を繰り返すことにより、符号化データ

(11, 10, 10, 11, 00, 11, 10)が得られる。

【0011】次に、この符号化データに対して3つ目の符号化データの第1ビット(第5ビット目)にエラーが発生し、

(11, 10, 00, 11, 00, 11, 10)

のデータが図14で示されるビタビ復号化装置131に伝送されたことを前提にビタビ復号化の説明を行う。ビタビ復号化処理は、ブランチメトリック処理と、ACS(加算→比較→選択)処理とトレースバック処理から構成されており、それぞれ図3(ブランチメトリック処理)、図17(ACS処理)、図18(トレースバック処理)のフローチャートで示されるような処理を行う。

【0012】(1)ブランチメトリック処理 受信信号がブランチメトリック処理部12に入力すると、ブランチメトリック処理部12では、図3に示すステップS31~S38において、時刻t[n-1]から時刻tnの1単位時間に遷移可能な全てのパスについて、ブランチメトリックを計算し、ACS処理部133に出力する。具体的には、時刻t1において符号化データ(1, 1)を受信すると、時刻t1の状態s0(図3の変数X=0)に遷移可能な前時刻t0のパスは状態s0、状態s2の2つであることが図11のトレリス線図より明らかである。

【0013】ここで、時刻t0の状態s0より遷移した場合、出力=(0, 0)であり、そのときは入力=0となる。同様に、時刻t0の状態s2より遷移した場合、出力=(1, 1)であり、そのときの入力=0であることが容易に推定できる(図3のステップS33)。これらの出力(00, 11)と受信した符号化データ=

(1, 1)よりブランチメトリックを求める(図3のステップS34)。ブランチメトリック計算では、前記出力(00, 11)と受信した符号化データ=(1, 1)のビットごとの差の総和を示すハミング距離(ブランチメトリック)を求める。

【0014】時刻t0の状態s0より遷移した場合は第1ビット、第2ビット共に一致しないのでハミング距離=2となる。同様に、時刻t0の状態s2より遷移した

場合は第1ビット、第2ビット共に一致するのでハミング距離=0となる。以上の処理を他の3つ(図3の変数 $X=1, 2, 3$)の状態全てについて行う(図3のステップS37、S38)。

【0015】(2) ACS(加算→比較→選択)処理
ブランチメトリック処理により計算したブランチメトリックがACS処理部133に入力すると、ACS処理部133では、図17に示すステップS71～S78において、このブランチメトリックと、それ以前に推定したバスとの関係に基づいて、最も確からしいバスを各状態毎に推定する。具体的には、時刻 t 1の状態 s 0(変数 $X=0$)に遷移可能な時刻 t 0の状態 s 0、状態 s 2におけるそれまでのバスメトリック(累積ハミング距離)と、ブランチメトリック処理部12で計算したブランチメトリック(ハミング距離)を加算し、時刻 t 1までのバスメトリックを計算する。この場合、時刻 t 0は初期状態であるため、累積ハミング距離=0として計算すると、上記の入力例では時刻 t 0の状態 s 0より遷移した場合はバスメトリック=2($=0+2$)、時刻 t 0の状態 s 2より遷移した場合はバスメトリック=0($=0+0$)となる(図17のステップS73)。

【0016】次に、それぞれのバスに対するバスメトリックを比較して、よりバスメトリックの小さい方の状態より遷移してきたと推定(バスメトリックが同一の場合はより状態番号の小さい方からと推定する)し、そのときの入力を復号データ候補とし、バスメモリ135における時刻 t 1の状態 s 0のエリアに・バスメモリへのポインタ・バスメトリック、および・復号データ候補を書き込む。

【0017】この場合、上記の入力例では時刻 t 0の状態 s 0より遷移した場合にはバスメトリック=2、時刻 t 0の状態 s 2より遷移した場合にはバスメトリック=0であるため、時刻 t 0の状態 s 2より時刻 t 1の状態 s 0に遷移したと推定し、バスメモリ135における時刻 t 1の状態 s 0のエリアに・バスメモリ135における時刻 t 0の状態 s 2のエリアへのポインタ・バスメトリック値=0・復号データ候補=0を書き込む(図17のステップS76)。以上の処理を他の3つ(図17の変数 $X=1, 2, 3$)の状態全てについて行う(図17のステップS73、S74)。

【0018】(3)トレースバック処理 各時刻において受信した符号化データについて、ブランチメトリック処理、ACS処理の一連の処理を行い、全ての受信信号(符号化データ)に対しての推論が終了した場合の状態遷移は図16のようになる。トレースバック処理部134では、図18に示すステップS81～S87において、最終時刻 t においてバスメトリックが最も小さい状態 s $X(t_n)$ から過去を選び、復号データ候補を求める。

【0019】具体的には、最終時刻 t_n ($n=7$)にお

ける全状態のバスメトリックを読み出し、バスメトリックが最も小さい状態 s 3を選択する。次に、その状態から推定したバスのバスメモリへのポインタを読み出すことを繰り返して過去へ遡って行くと(図18のステップS85)と、図16の実線で示される状態を遷移してきたことが推定できる。この状態遷移は図13の状態遷移と同一であり、この状態遷移の復号データ候補(図16中の網付きのボックス)を過去から順番に取り出すと(1, 1, 0, 0, 0, 1, 1)

となり、符号化される前のデータと一致することがわかる。以上説明したように、ビタビ復号化では、伝送されてきた符号化データに含まれるエラーを訂正し、正しいデータに復号することが可能となる。

【0020】なお、上記の従来例の説明においては、符号化データ列が短い場合(11, 10, 00, 11, 00, 11, 10の14ビット)で説明したが、一般的な符号化データはかなり長いため、厳密にビタビ復号化を行う場合にはバスメモリ135の容量およびトレースバック処理の時間が膨大になってしまう。このため、実際のビタビ復号化装置では、バスメモリ135は任意の長さで構成し、現時刻からバスメモリ135の長さ分の最新のバスを記憶しておき、新しいバスを書き込むときにトレースバック処理を行い、最も古いバスの復号データ候補を出力後、最も古いバスを捨てるようにしている。一般的にバスメモリ135の長さ(トレースバック長)は符号化装置の拘束長の5～6倍にすることで、打ち切りの誤差をほぼ無視できることが知られている。

【0021】他の従来例としては、例えば特開平5-55931号公報に示すように回路規模を小さくするためにバスメトリック値を少ないビット数で表すとともに、高速化するために正規化処理を毎回ではなく、バスメトリックがオーバーフローしない程度の周期で行う方法が提案されている。また、他の従来例としては、例えば特開平6-77845号公報に示すように伝送効率を向上させるために複数のビタビ復号器を組み合わせる方法が提案されている。また、特開平6-197027号公報には、エラーレートを向上させるための方法が提案されている。また、特開平8-167858号公報には、ACS処理後、 n 個のバッファと分配器でバスメモリを動作させる構成が提案されている。また、特開平9-51278号公報には、1つのバスメモリの代わりに $1/n$ の大きさのバッファを n 個用いて独立して動作させる構成が提案されている。

【0022】

【発明が解決しようとする課題】従来のトレースバック法によるビタビ復号化方法の第1の問題点は、1回の復号化処理におけるバスメモリ135へのメモリアクセス回数が増大し、処理性能が低下することである。その理由は、例えば、トレースバック長=16個(実際のビタビ復号化で用いられる一般的な値)とすると、トレース

バック処理において、到達時刻の全状態のバスメトリック値の読み出しと、1時刻前に遡るために「推定したバスのバスメモリへのポインタの読み出し」がトレースバック長分だけ発生するため、 $4 + 16 = 20$ 回のバスメモリ135に対するメモリアクセスが発生する。この回数は符号化装置の拘束長が長いほど増大することになる。近年、マイクロプロセッサの動作速度は高速化しているが、実際のシステムで使用されるメモリのアクセス時間は遅いため（コストとのトレードオフ）、メモリアクセスの回数が増えることはそのまま処理性能を低下させる原因となってしまう。

【0023】第2の問題点は、メモリ（バスメモリ135）容量の増加である。その理由は、バスメモリ135の容量がトレースバック長に依存しているためである。前記条件（トレースバック長=16、状態数=4）でのバスメモリ135の容量は、 $3 \times 4 \times 16 = 192$ （ワード）となる。この値は一般的なメモリを想定した場合は特に問題とはならないが、マイクロプロセッサに内蔵されているメモリ（キャッシュメモリなど）では大きな問題となってしまう（キャッシュ・ミスヒット増加による処理性能の低下）。

【0024】

【発明の目的】本発明は上記従来例の問題点に鑑み、トレースバック処理時におけるバスメモリへのメモリアクセス回数により処理性能が低下することを防止することができ、また、バスメモリの容量を低減することができるビタビ復号化装置およびビタビ復号化方法並びにコンピュータプログラム記録媒体を提供することを目的とする。

【0025】

【課題を解決するための手段】本発明のビタビ復号化装置は、上記目的を達成するために、前時刻から現時刻までの1単位時間に遷移可能な全てのバスについてブランチメトリックを計算するブランチメトリック処理手段と、前時刻までの最尤復号情報と現時刻までの最尤復号情報をそれぞれ記憶するための1対のデータバッファと、前時刻までのバスメトリックと現時刻までのバスメトリックをそれぞれ記憶するための1対のバスメトリックバッファと、1単位時間毎に、前記ブランチメトリック処理手段により計算されたブランチメトリックと前記1対のバスメトリックバッファの一方に記憶された前時刻までのバスメトリックに基づいて、現時刻の各状態における最も確からしいバスを推定して、このバスに基づいて前記1対のデータバッファの一方に記憶された前時刻までの最尤復号情報を並び替えて他方に書き込むとともに、前記1対のバスメトリックバッファの一方に記憶されたバスメトリックを更新して他方に書き込むACS処理手段と、前記バスメトリックバッファに記憶された現時刻までのバスメトリックと前記データバッファに記憶された現時刻までの最尤復号情報に基づいて復号デー

タを検索するトレースバック処理手段とを有することを特徴とする。

【0026】上記構成により、ACS処理の単位毎に、データバッファに記憶される最尤復号情報とバスメトリックバッファに記憶されるバスメトリックが更新される。このため、トレースバック処理では現時刻までのバスメトリックに基づいて現時刻までの最尤復号情報をアクセスするのみで復号することができる。

【0027】また、データバッファには、最尤復号情報として各バス毎の最尤復号データ候補や最尤状態遷移履歴が記憶される。

【0028】

【発明の実施の形態】以下、図面を参照して本発明の実施の形態を説明する。図1は本発明に係るビタビ復号化装置の一実施形態を示すブロック図、図2は図1のデータバッファおよびバスメトリックバッファを示す構成図である。

【0029】図1を要約すると、本発明に係るビタビ復号化装置11は、図2に示すように復号データ候補を格納するための1対の同一構成のデータバッファ17a、17bと、バスメトリックを記憶する1対の同一構成のバスメトリックバッファ18a、18bを有する。このデータバッファ17a、17bの一方には前時刻までの復号データ候補が格納され、他方には現時刻の復号データ候補が格納される。この場合、ACS（Add Compare Selection）処理において、データバッファ17a、17bの更新時に前時刻までの復号データ候補が格納されているデータバッファ17a、17bの一方から、推定された前状態の情報が格納されているラインを読み出し、データバッファ17a、17bの他方の現状態のラインに対して、推定された復号データ候補を付加して書き込む。

【0030】上記処理を各時刻の各状態毎に行うことにより、データバッファ17a、17bの各ラインは各状態における最尤バスの復号データ候補系列となり、トレースバック処理において、従来と異なり1時刻ずつ遡るのではなく、直接任意の時刻まで遡ることが可能となり、これによりトレースバック処理時間を短縮することができる。

【0031】詳しく説明すると、ビタビ復号化装置11は、バスメモリ15と、前時刻から現時刻までの1単位時間に遷移可能な全てのバスについてブランチメトリックを計算してこれをACS処理部13に出力するブランチメトリック処理部12と、前記ブランチメトリック処理部12で計算されたブランチメトリックと、バスメモリ15内のセクタ16により選択される前時刻までのバスメトリックバッファ18a、18bに記憶されたバスメトリックに基づいて、現時刻の各状態における最も確からしいバスを推定し、復号データ候補を並び替えるACS処理部13と、バスメモリ15内のセクタ16

により選択される現時刻のバスメトリックバッファ18 a、18 bに記憶されたバスメトリックと、現時刻のデータバッファ17 a、17 bに記憶された復号データ候補に基づいて、出力すべき復号データを検索するトレースバック処理部14より構成されている。

【0032】バスメトリック15は、図2に示すように状態数×トレースバック長分の数の復号データ候補を記憶する1対の同一構成のデータバッファ17 a、17 bと、状態数分のバスメトリックを記憶する1対の同一構成のバスメトリックバッファ18 a、18 bと、1単位時間毎にバッファ(17 a、17 b)、(18 a、18 b)の切り替えを行うセクタ16とを有する。

【0033】図3は本発明のブランチメトリック処理を示すフローチャート、図4および図5は本発明のACS処理を示すフローチャート、図6はデータバッファ17 a、17 bの動作を示す説明図、図7はデータバッファ17 a、17 bとバスメトリックバッファ18の遷移を示す説明図、図8は本発明のトレースバック処理を示すフローチャートである。

【0034】なお、説明においては、時間の流れを1単位時間毎に t_1 、 t_2 、 \dots 、 t_n とし、各時刻の1単位時間前を $t[n-1]$ とした(時刻 t_3 の1単位時間前は、時刻 t_2)。また、セクタ16により切り替えられるa側のバッファ群17 a、18 aは時刻 t_1 、 t_3 、 \dots の時はリード動作、時刻 t_2 、 t_4 、 \dots の時はライト動作を行い、b側のバッファ群17 b、18 bは時刻 t_1 、 t_3 、 \dots の時はライト動作、時刻 t_2 、 t_4 、 \dots の時はリード動作を行うこととし、ビタビ復号化する対象の符号化データ(符号化前のデータは1、1、0、0、0、1、1)は、従来例で説明したものと同一データ(11、10、00、11、00、11、10)であることを前提に説明を行う。

【0035】(1)ブランチメトリック処理 受信信号がビタビ復号化装置11のブランチメトリック処理部12に入力すると、ブランチメトリック処理部12では、図3に示すステップS31～S38において時刻 $t[n-1]$ から時刻 t_n の1単位時間に遷移可能な全てのパスについてブランチメトリックを計算し、ACS処理部13に出力する。具体的には、時刻 t_1 において符号化データ=(1, 1)を受信すると、時刻 t_1 の状態 s_0 に遷移可能な前時刻 t_0 のパスは状態 s_0 、状態 s_2 の2つであることが図12に示すトレース線図より明らかである。ここで、時刻 t_0 の状態 s_0 より遷移した場合、出力=(0, 0)であり、そのときの入力=0となる。同様に、時刻 t_0 の状態 s_2 より遷移した場合、出力=(1, 1)であり、そのときの入力=0であることが容易に推定できる(図3のステップS33)。

【0036】これらの出力(00, 11)と受信した符号化データ=(1, 1)よりブランチメトリックを求め(図3のステップ34)。ブランチメトリック計算で

は、前記出力(00, 11)と受信した符号化データ=(1, 1)のビットごとの差の総和を示すハミング距離を求める。時刻 t_0 の状態 s_0 より遷移した場合は第1ビット、第2ビット共に一致しないのでハミング距離=2となる。同様に、時刻 t_0 の状態 s_2 より遷移した場合は第1ビット、第2ビット共に一致するのでハミング距離=0となる。以上の処理を他の3つ(図3の変数 $X=1, 2, 3$)の状態全てについて行う(図3のステップS37、S38)。

10 【0037】(2)ACS処理 ブランチメトリック処理において計算したブランチメトリックは、ACS処理部13に入力し、ACS処理部13では図に示すステップS41～S61において、また、図6、図7に示すようにブランチメトリックとそれ以前に推定したパスとの関係に基づいて、最も確からしいパスを各状態毎に推定する。具体的には、時刻 t_1 の状態 s_0 に遷移可能な時刻 t_0 の状態 s_0 、状態 s_2 におけるそれまでのバスメトリック(累積ハミング距離)と、ブランチメトリック処理部12で計算したブランチメトリック(ハミング距離)を加算し、時刻 t_1 までのバスメトリックを計算する。

【0038】この場合、時刻 t_0 のバスメトリックの時には、セクタ16によりa側のバスメトリックバッファ18 aが選択される。このとき、バスメトリックバッファ18 aは初期状態であるためバスメトリック=0として計算すると、時刻 t_0 の状態 s_0 より遷移した場合はバスメトリック=2(=0+2)、時刻 t_0 の状態 s_2 より遷移した場合はバスメトリック=0(=0+0)となる(図4のステップS44)。

30 【0039】次に、それぞれのパスに対するバスメトリックを比較して、よりバスメトリックの小さい方の状態より遷移してきたと推定(バスメトリックが同一の場合はより状態番号の小さい方からと推定する)し、そのときのバスメトリックをb側のバスメトリックバッファ18 bに書き込む(図5のステップS47)。この場合、上記の入力例では時刻 t_0 の状態 s_0 より遷移した場合のバスメトリック=2、時刻 t_0 の状態 s_2 より遷移した場合のバスメトリック=0であるため、時刻 t_0 の状態 s_2 より時刻 t_1 の状態 s_0 に遷移したと推定し、b側のバスメトリックバッファ18 bの状態 s_0 にバスメトリック値=0を書き込む。

40 【0040】次に、a側のデータバッファ17 aから、推定された状態 s_2 の復号データ候補系列を読み出し(図5のステップS48)、推定されたパス($s_2 \rightarrow s_0$)により確定する復号データ候補=0を付加して、b側のデータバッファ17 bの状態 s_0 の復号データ候補系列に書き込む(図5のステップS49、図6の時刻 t_1)。以上の処理を他の3つ(図4の変数 $X=1, 2, 3$)の状態全てについて行う(図5のステップS50、S51)。

【0041】次の時刻 t_2 において、符号化データ=(1, 0)を受信したとき、ブランチメトリック処理後、時刻 t_2 の状態 s_0 に遷移可能な時刻 t_1 の状態 s_0 、状態 s_2 におけるそれまでのバスメトリックと、ブランチメトリック処理部12で計算したブランチメトリックを加算し、時刻 t_2 までのバスメトリックを計算する。この場合、時刻 t_1 のバスメトリックはセクタ16により b 側のバスメトリックバッファ18 b が選択される。このとき、バスメトリックバッファ18 b の状態 s_0 のバスメトリック=0であるため、時刻 t_1 の状態 s_0 より遷移した場合はバスメトリック=1(=0+1)、時刻 t_1 の状態 s_2 より遷移した場合は、バスメトリック=2(=1+1)となる。

【0042】次に、それぞれのバスに対するバスメトリックを比較して、よりバスメトリックの小さい方の状態より遷移してきたと推定し、そのときのバスメトリックを a 側のバスメトリックバッファ18 a に書き込む。この場合、上記の入力例では時刻 t_1 の状態 s_0 より遷移した場合のバスメトリック=1、時刻 t_1 の状態 s_2 より遷移した場合のバスメトリック=2であるため、時刻 t_1 の状態 s_0 より時刻 t_2 の状態 s_0 に遷移したと推定し、 a 側のバスメトリックバッファ18 a の状態 s_0 にバスメトリック値=1を書き込む。

【0043】次に、 b 側のデータバッファ17 b から、推定された状態 s_0 の復号データ候補系列を読み出し、推定されたバス($s_0 \rightarrow s_0$)により確定する復号データ候補=0を付加して、 a 側のデータバッファ17 a の状態 s_0 の復号データ候補系列に書き込む(図6の時刻 t_2)。以上の処理を他の3つ(図4の変数 $X=1, 2, 3$)の状態全てについて行う。

【0044】以上のような動作を繰り返すことにより、セクタ16により選択される現時刻 t_n のデータバッファ17の1ラインは各状態における最尤バスの復号データ候補系列となる。つまり、1ラインの復号データ候補系列の中で最も古いデータが、現時刻 t_n における復号データとなる。

【0045】ここで、符号化データとして(11, 10, 00, 11, 00, 11, 10)が入力されたときの復号動作におけるデータバッファ17とバスメトリックバッファ18の遷移を図7に示す。図7を参照すると、最終時刻 t_7 における b 側のバスメトリックバッファ18 b を見ると状態 s_3 のバスメトリックが最小値=1であることから、 b 側のデータバッファ17 b の状態 s_3 の1ラインに記憶されている復号データ候補系列を、最も古い時刻 t_1 から順にならべると(1, 1, 0, 0, 0, 1, 1)となり、符号化される前のデータと一致していることがわかる(正しいデータに復号)。

【0046】(3)トレースバック処理 トレースバック処理部14では、前記ACS処理により、データバッファ17の各ラインは各状態における最尤復号データ系

列となるため、従来と異なり1単位時間づつ遡ることなく、図8に示すステップS61~S64において直接トレースバック長分の過去まで遡ることができる。

【0047】このため、第1の効果は、トレースバック処理の時間を短縮できることである。その理由は、トレースバック処理において、直接最も古い復号データ候補を検索することができるため、1時刻づつ過去へ遡るためのメモリアクセスが不要となるからである。具体的には、例えばトレースバック長=16とすると、従来では、 $4+16=20$ 回のメモリアクセスが必要であったのに対し、本発明によれば、 $4+1=5$ 回のメモリアクセスでトレースバック処理を行うことができる。

【0048】ここで、ビタビ復号化(トレースバック長=16)処理を実際にマイクロプロセッサ(NEC μ PD705100:V830)を使用してプログラミングした時の性能データを比較してみると、従来方法では $90\mu s$ (周期処理単位: $416\mu s$)かかっていた処理を本発明のビタビ復号化方法により、 $45\mu s$ に短縮することができた。また、本発明によればトレースバック長に依存せずに、トレースバック処理を行うことができるため、トレースバック長が長くなるほど効果が高くなることは明らかである。

【0049】第2の効果は、バスメモリ容量を低減できることである。その理由は、バスメモリを一对のバッファ群で構成することにより、不要なバスメトリック用のメモリを削減できるからである。具体的には、例えば、トレースバック長=16、状態数=4とすると、従来、 $3 \times 4 \times 16 = 192$ (ワード)のバスメモリが必要であったのに対し、本発明によれば、 $(4 + 4 \times 16) \times 2 = 136$ (ワード)のメモリ容量でビタビ復号化処理を行うことができる。このため、キャッシュメモリを内蔵したマイクロプロセッサではキャッシュ・ヒット率が向上し、処理時間をさらに短縮することができる。

【0050】なお、上記実施形態では、データメモリに対して復号データ候補を記憶したが、代わりに最尤状態遷移履歴を記憶するようにしてもよい。

【0051】

【発明の効果】以上説明したように本発明によれば、ACS処理の単位毎に、データバッファに記憶される最尤復号情報とバスメトリックバッファに記憶されるバスメトリックが更新されるので、トレースバック処理時におけるバスメモリへのメモリアクセス回数により処理性能が低下することを防止することができ、また、バスメモリの容量を低減することができる。

【図面の簡単な説明】

【図1】本発明に係るビタビ復号化装置の一実施形態を示すブロック図である。

【図2】図1のデータバッファおよびバスメトリックバッファを示す構成図である。

【図3】本発明のブランチメトリック処理を示すフロー

チャートである。

【図 4】本発明の ACS 処理を示すフローチャートである。

【図 5】本発明の ACS 処理を示すフローチャートである。

【図 6】図 1 のデータバッファの動作を示す説明図である。

【図 7】図 1 のデータバッファとバスメトリックバッファの遷移を示す説明図である。

【図 8】本発明のトレースバック処理を示すフローチャートである。

【図 9】畳み込み符号化装置を示すブロック図である。

【図 10】図 9 の装置の状態を示す説明図である。

【図 11】図 9 の装置の状態遷移を示す説明図である。

【図 12】図 9 の装置のトレリス線図を示す説明図である。

【図 13】図 9 の装置の状態遷移の一例を示す説明図である。

*【図 14】従来のビタビ復号化装置を示すブロック図である。

【図 15】図 14 のバスメモリを詳細に示すブロック図である。

【図 16】トレースバック法による復号動作における状態遷移を示す説明図である。

【図 17】従来の ACS 処理を示すフローチャートである。

【図 18】従来のトレースバック処理を示すフローチャートである。

【符号の説明】

12 ブランチメトリック処理部

13 ACS 処理部

14 トレースバック処理部

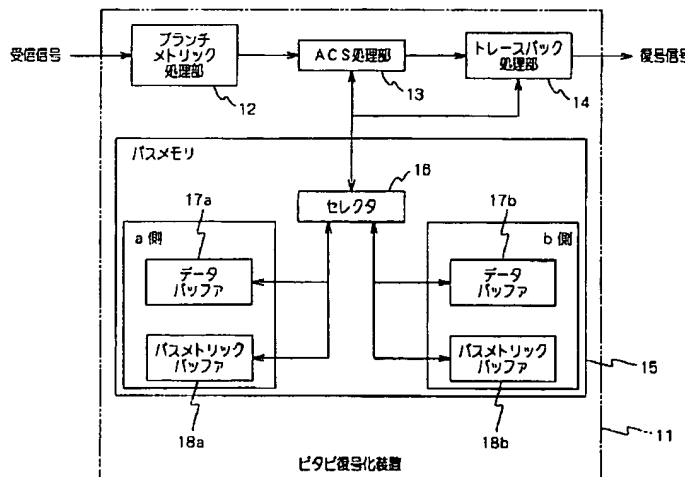
15 バスメモリ

16 セレクタ

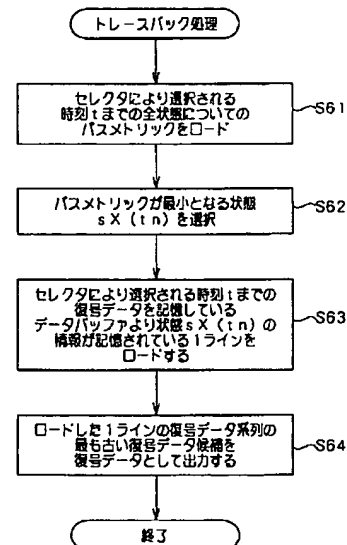
17 a, 17 b データバッファ

* 18 a, 18 b バスメトリックバッファ

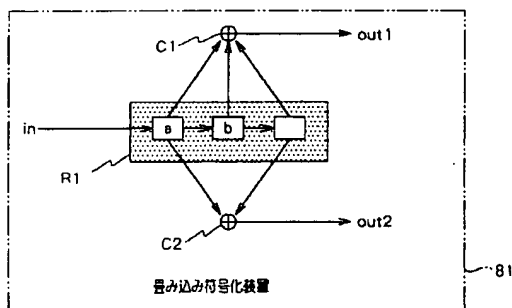
【図 1】



【図 8】



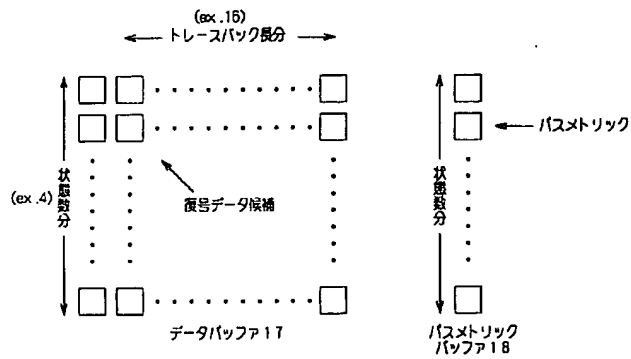
【図 9】



【図 11】

現在の状態	入力(in)	出力(out2, out1)	遷移する状態
s0	0 1	00 11	s0 s1
s1	0 1	01 10	s2 s3
s2	0 1	11 00	s0 s1
s3	0 1	10 01	s2 s3

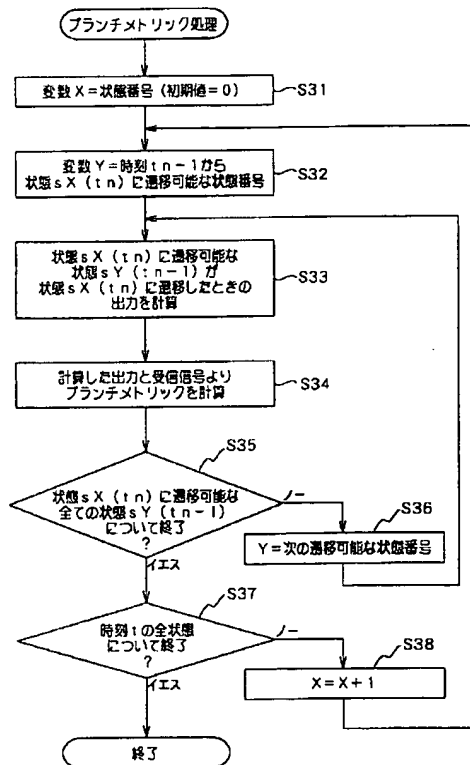
【図2】



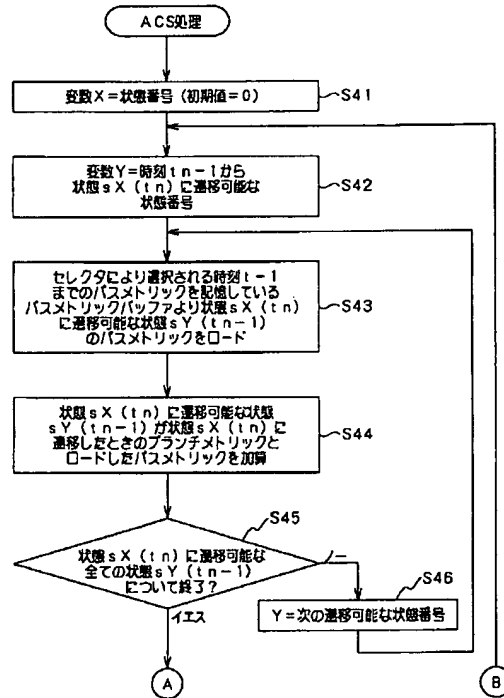
【図10】

a	b	状態
0	0	S 0
0	1	S 1
1	0	S 2
1	1	S 3

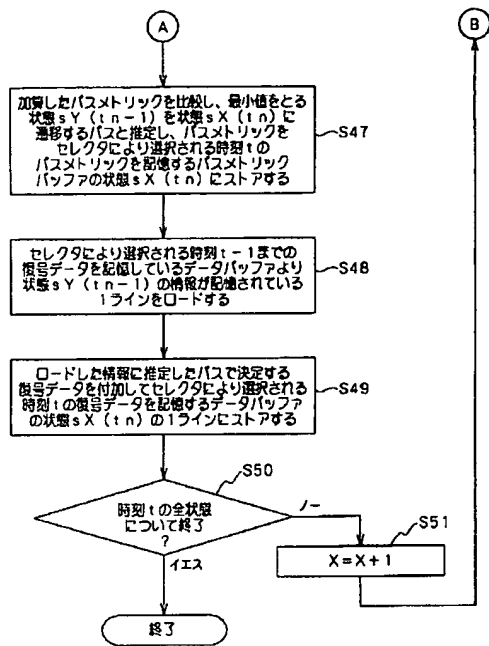
【図3】



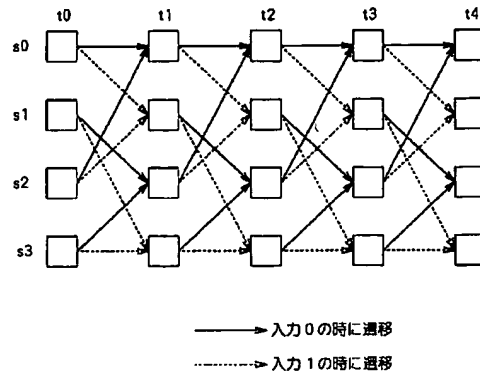
【図4】



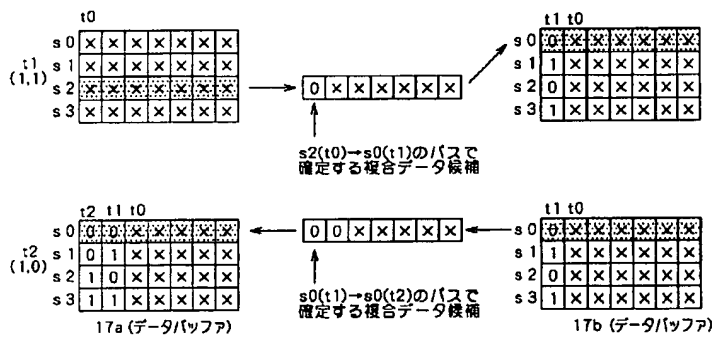
【図5】



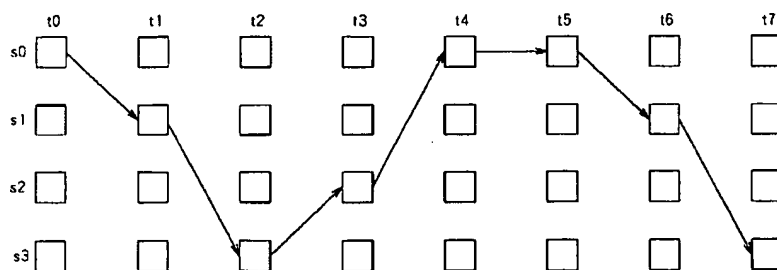
【図12】



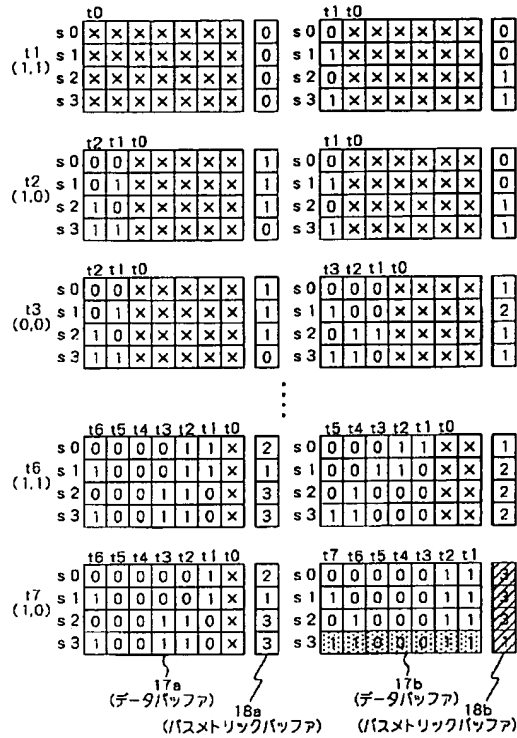
【図6】



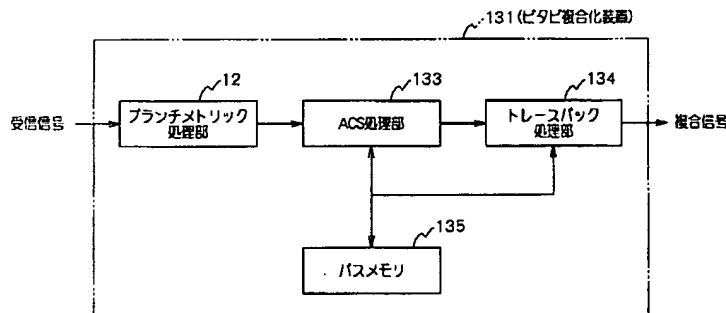
【図13】



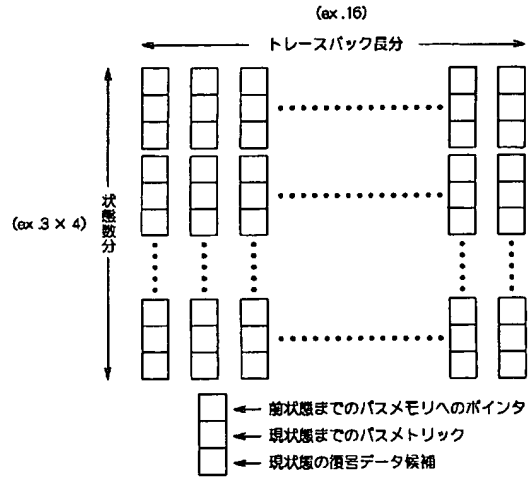
【図7】



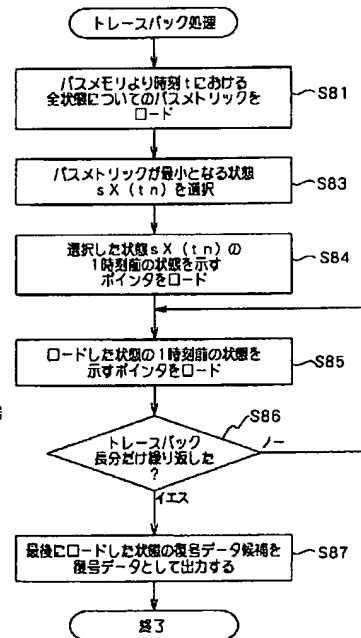
【図14】



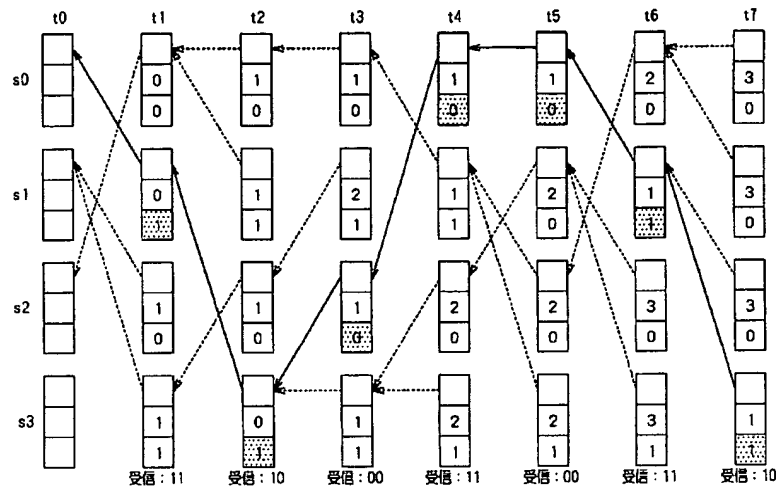
【図15】



【図18】



【図16】



【図17】

